National Research Council Canada
Conseil national de recherches Canada

Division of Electrical Engineering
Division de génie électrique

ERB-987

Computer Programs for L_1 Polygonal Approximation of Planar Curves

Nabih N. Abdelmalek

March 1986

Canada

NRCC No. 25078

85-04299/

Copyright 1986 by
National Research Council of Canada

Permission is granted to quote short excerpts and to reproduce figures and tables from this report, provided that the source of such material is fully acknowledged.

Copyright 1986 par
Conseil national de recherches du Canada

Il est permis de citer de courts extraits et de reproduire des figures ou tableaux du présent rapport, à condition d'en identifier clairement la source.

Additional copies are available free of charge from:

Editorial Office, Room 301
Division of Electrical Engineering
National Research Council of Canada
Ottawa, Ontario, Canada
K1A 0R8

Des exemplaires supplémentaires gratuits peuvent être obtenus à l'adresse suivante:

Bureau des publications, Pièce 301
Division de génie électrique
Conseil national de recherches du Canada
Ottawa (Ontario) Canada
K1A 0R8

COMPUTER PROGRAMS FOR L_1 POLYGONAL APPROXIMATION OF PLANAR CURVES

Nabih N. Abdelmalek

ERB-987

NRCC No. 25078

MARCH 1986

TABLE OF CONTENTS

	Page
ABSTRACT/RÉSUMÉ	v
1. INTRODUCTION	1
2. PRELIMINARIES AND NOTATION	1
3. THE L_1 APPROXIMATION PROBLEM	2
4. THE ALGORITHM	3
4.1 A Linear Programming Technique	4
4.2 The Algorithm using Linear Programming	5
5. THE COMPUTER PROGRAMS	7
6. NUMERICAL RESULTS	7
7. COMMENTS AND CONCLUSION	9
ACKNOWLEDGEMENTS	10
REFERENCES	10
APPENDIX	13
Driver DR1L1POL	13
Subroutine PC1L1POL	17
Subroutine LONEPOL	21

FIGURES

1. Polygonal approximation for example 1, the L_1 norm in any segment is ≤ 1.0	7
2. Polygonal approximation for a wave form, the L_1 norm in any segment is ≤ 1.0	8
3. Polygonal approximation for a contour, the L_1 norm in any segment is ≤ 0.5	8
4. Polygonal approximation for a self intersecting curve, the L_1 norm in any segment is ≤ 0.8	9

ABSTRACT

A computer subroutine and its subsidiary routine, written in FORTRAN 77, for obtaining a polygonal approximation in the L_1 norm of a plane curve of arbitrary shape are given. The L_1 error norm in any segment is not to exceed a pre-assigned value. The subroutine is written for an algorithm described in a recent paper by the author. The given curve is first digitized and the algorithm is then applied to the discrete points. The algorithm uses linear programming techniques which make it an efficient and a fast one. Numerical results and comments are given. Given as well is a Driver for the given subroutine. Numerical examples are presented.

RÉSUMÉ

Ce rapport décrit une sous-routine d'ordinateur et sa routine subsidiare, écrites en FORTRAN 77, pour l'approximation polygonal de courbe plane de figure arbitraire avec la norme L_1. L'erreur de norme L_1 pour chaque segment ne doit pas excède une valeur pré-déterminée. La sous-routine est écrite pour un algorithme décrit dans un article récent du même auteur. La courbe est d'abord numérisée et l'algorithme appliqué aux points ainsi relevés. L'algorithm est basé sur des techniques de programmation lineaire donnant une exécution efficase et rapide. Un programme de démonstration pour cette sous-routine est aussi inclu, ainsi des examples numériques.

COMPUTER PROGRAMS FOR L_1 POLYGONAL APPROXIMATION OF PLANAR CURVES

Nabih N. Abdelmalek

1. INTRODUCTION

Piecewise approximation of plane curves has several applications. In the fields of image processing and pattern recognition, it is used in feature extraction, noise filtering, and data compression (1). It is also used in speech recognition and numerous other fields.

In the presence of spikes and wild points in a given curve, the L_1 approximation to the given curve is usually recommended over other norms (2). To our knowledge the problem of piecewise approximation in the L_1 norm has received very little attention. Recently the author (3, 4) presented two algorithms for piecewise linear L_1 approximation of planar curves. Using a geometric method, Johnson and Vogt (5) approximated a planar convex curve by a sequence of n points along the curve. Their method is a generalization to a method which minimizes the area between the curve and the polygonal arc having the n points as its successive vertices. This method is a kind of an L_1 piecewise approximation method.

In the present work, a computer subroutine written in FORTRAN 77 for obtaining a polygonal approximation in the L_1 norm of a plane curve of arbitrary shape is presented. The L_1 error norm in any segment is not to exceed a pre-assigned value. The subroutine is written for an algorithm described in a recent paper by the author (6). The problem is solved by first digitizing the given curve. Then the algorithm is applied to the discrete points. The vertices of the polygonal arc are a subset of the points on the given digitized curve. In other words, the approximating straight line to any segment of the given curve passes through the two end points of the segment. The algorithm uses linear programming techniques (7), making it efficient and fast.

For piecewise and polygonal approximations in the least squares and in the Chebyshev norms, see the references cited in Abdelmalek (8) and in Kurozumi and Davis (9). See also Sklansky and Gonzalez (10) and Gritzali and Papakonstantinou (11).

2. PRELIMINARIES AND NOTATION

Let us be given a two-dimensional open or closed curve. Let this curve be digitized and let the digitized set be ordered at K consecutive points.

The first and the last points coincide for a closed curve. The digitized points may not be equally spaced. Let n be the number of pieces (segments) in the approximation and let $\{z_j\}$, $j = 1, \ldots, n$, be the L_1 norms for the n pieces; n is not known before hand.

3. THE L_1 APPROXIMATION PROBLEM

Consider any segment j, $1 \leq j \leq n$, of the given curve. Let this segment consist of N digitized points, with coordinates $\{x_i, f(x_i)\}$, $i = 1, \ldots, N$, $N > 2$. Let these N points be approximated by the straight line

$$[1] \qquad y = a_1 + a_2 x$$

which minimizes the L_1 norm z_j (or briefly z), of the residuals $\{r_i\}$, where

$$[2a] \qquad z = \sum_{i=1}^{N} |r_i|$$

$$[2b] \qquad r_i = a_1 + a_2 x_i - f_i \,, \quad i = 1, \ldots, N$$

and f_i denotes $f(x_i)$.

It is known that this problem reduces to the problem of obtaining the L_1 solution of the following overdetermined system of linear equations (12).

$$[3a] \qquad \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ . & . \\ . & . \\ . & . \\ 1 & x_N \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ . \\ . \\ . \\ f_N \end{bmatrix}$$

or

$$[3b] \qquad Ca = f$$

C is the coefficient matrix in Eq. [3a], vector $a = (a_1, a_2)^T$ and vector $f = (f_1, f_2, \ldots, f_N)^T$. The superscript T denotes the transpose.

The L_1 solution to system [3] is the real vector a^* which minimizes the norm z given by [2a].

As mentioned at the end of Section 1 above, the approximating straight line to segment j interpolates the end points (points 1 and N) of segment j.

Hence this problem is not a minimization one, since we have two unknowns (a_1, a_2) and two conditions to be fulfilled; interpolating the two end points. However, if the approximating curve has more than two parameters, for example, a vertical parabola of the form

$$[4] \qquad y = a_1 + a_2x + a_3x^2$$

the problem would be an optimization problem as follows. Obtain the L_1 solution of the overdetermined system of linear equations [3b], subject to the conditions that the curve [4] interpolates the two end points of the segment at hand. In this case matrix C in Eq. [3b] is an N by 3 matrix whose first two columns are the same as the coefficient matrix in Eq. [3a] and whose third column is $(x_1^2, x_2^2, \ldots, x_N^2)^T$, and vector $a = (a_1, a_2, a_3)^T$.

4. THE ALGORITHM

Let the piecewise approximation for any segment j, $1 \leq j \leq n$, be such that its norm $z \leq \varepsilon$, where ε is a pre-assigned value. The outline of this algorithm is as follows:

Step 1. Take j = 1 (the first segment).

Step 2. Take N = 2, which is the minimum initial number of points for segment j to be approximated by the straight line [1]. Calculate the solution (a_1, a_2) for the line which passes through the two points. The norm z = 0, since it corresponds to a perfect fit to the two points. Go to step 3.

In case points 1 and N lie on a vertical or a nearly vertical line, a_2 (the slope of the line), would be a very large number and the computation may break down. In this case, increase the number of points N by 1, if necessary more than once, until the end points 1 and N are not on a vertical or a nearly vertical line.

Step 3. Add the point (N + 1) of the curve to segment j and calculate the solution (a_1, a_2) for the straight line which passes through the points 1 and (N + 1). Calculate z from [2a]. Take N = N + 1. Go to step 4.

Again as explained above, if points 1 and (N + 1) lie on a vertical or a nearly vertical line, increase the number of points by 1, if necessary more than once.

Step 4. If $z \leq \varepsilon$; the pre-assigned norm, go to step 3. If $z > \varepsilon$, take N = N - 1, which is the final number of points for segment j, with the

corresponding solution (a_1, a_2) as the required L_1 solution for segment j. Take j = j + 1 and go to step 2 for segment (j + 1).

Since this algorithm is for a polygonal approximation, the last end point of segment j, is itself the first end point of segment (j + 1).

4.1 A Linear Programming Technique

We now implement this algorithm using linear programming techniques. This enables us to calculate in an efficient way the norm z each time we add a new point to the N points of segment j. This is done by simply updating a simplex tableau by applying (Gauss-Jordan) elimination steps.

Let matrix C^T form the initial data of a linear programming problem and vector f be the cost vector (7) to the initial data, where C and f are those of Eq. [3b] above and N > 2. Let a 2 by 2 unit matrix I_2 be considered as the inverse of an initial basis matrix I_2; denoted by B_0^{-1}. This is shown on the left of Tableau 1.

We update Tableau 1 (see below), by applying two elimination steps which reduce columns 1 and N to the first and second columns of the 2 by 2 unit matrix I_2 respectively. We get Tableau 2.

Tableau 1

B_0^{-1}	f_1	f_2	f_2	.	.	.	f_N
$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	1	1	.	.	.		1
	x_1	x_2	.	.	.		x_N

Tableau 2

B_1^{-1}	f_1	f_2	f_3	.	.	.	f_N
$\begin{pmatrix} 1 & 1 \\ x_1 & x_N \end{pmatrix}^{-1}$	1	y_{21}	y_{31}	.	.	.	0
	0	y_{22}	y_{31}	.	.	.	1
	0	c_2	c_3				0

If points 1 and N lie on a vertical line, the pivot element for the second elimination step in Tableau 1, would be zero or a very small number and the computation would break down. See Lemma 3 below. In this case, as mentioned earlier, we increase the number of points N by 1, if necessary more than once until the pivot element for the added point is greater than a specified tolerance.

On the left of Tableau 2, $B_1^{-1} = \begin{pmatrix} 1 & 1 \\ x_1 & x_N \end{pmatrix}^{-1}$ is the inverse of the basis matrix for Tableau 2. It is obtained by merely updating B_0^{-1} $(=I_2)$ shown with Tableau 1, as if B_0^{-1} were a part of Tableau 1.

In Tableau 2, the vectors $\{y_i\}$, i = 1, ..., N, are given by

$$[5] \qquad y_i = B_1^{-1} \begin{pmatrix} 1 \\ x_i \end{pmatrix}, \quad i = 1, \ldots, N$$

The elements of the last row in Tableau 2, known as the the marginal costs, are given by

$$[6] \qquad c_i = f_1 y_{i1} + f_N y_{i2} - f_i, \quad i = 1, \ldots, N$$

Tableau 2 corresponds to the end of step 2 in Section 3 above and also to the end of step 2 in Section 4.2 below.

Lemma 1: (Abdelmalek (13), Lemma 2, p. 44). The marginal costs $\{c_i\}$, i = 1, ..., N, in any tableau are themselves the residuals given by [2a];

$$[7] \qquad c_i = r_i, \quad i = 1, \ldots, N$$

and since $r_1 = r_N = 0$,

$$[8] \qquad z = |c_2| + |c_3| + \ldots + |c_{N-1}|$$

Lemma 2: The solution vector 'a' of the interpolating straight line is given by

$$[9] \qquad (a_1, a_2) = (f_1, f_N)\, B^{-1}$$

See the proof of Lemma 2, in ref. 13, p. 44.

4.2 The Algorithm Using Linear Programming

Step 1. Take j = 1 (the first segment).

Step 2. Take N = 2. Form Tableau 1 and Tableau 2 above respectively for the two points. For N = 2, z = 0. Go to step 3.

Step 3. Add the point (N + 1) of the curve to segment j. From [5] and [6] calculate y_{N+1} and c_{N+1}, respectively. Append y_{N+1}, f_{N+1}, and c_{N+1} to the right of Tableau 2. Apply an elimination step

in Tableau 2 which reduces y_{N+1} to the second column of the unit matrix I_2 and also reduces c_{N+1} to zero. Take N = N + 1. Go to step 4.

Step 4. Calculate the norm z from [8]. If $z \leq \varepsilon$, go to step 3. If $z > \varepsilon$, take N = N - 1, which is the final number of points for segment j. From [9] calculate the solution vector (a_1, a_2) as the L_1 solution for segment j. Take j = j + 1 and go to step 2 for segment (j + 1).

Notice in this subsection that the solution vector 'a' is calculated once only, which is at the end of Step 4.

Lemma 3: If a new added point (N + 1) lies on a vertical line with point 1 of the segment at hand, then $y_{N+1} = (1, 0)^T$ and the next elimination step breaks down.

Proof: If the points 1 and (N + 1) lie on a vertical line, then $x_{N+1} = x_1$ and in a straightforward manner, from [5],

$$y_{N+1} = B^{-1} \begin{pmatrix} 1 \\ x_{N+1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ x_1 & x_N \end{pmatrix}^{-1} \begin{pmatrix} 1 \\ x_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Assuming now that the points 1 and (N + 1) lie on a vertical line, before the next elimination step, one appends y_{N+1}, f_{N+1}, and c_{N+1} to the right of Tableau 2. In this elimination step, vector y_{N+1} is to be reduced to the second column of I_2. This would not be possible as the pivot element, the second element of y_{N+1}, is zero and the calculation breaks down.

If points 1 and (N + 1) lie on a nearly vertical line, the pivot element would be very small and again the calculation may break down.

Hence if the magnitude of a pivot element is <EPS, where EPS is a specified tolerance, we increase the number of points by one, if necessary more than once, until the magnitude of the pivot element of the added point is >EPS. For the VAX 11/780 computer, the round-off level is about 10^{-6} for single precision and about 10^{-16} for double precision. For this computer we take EPS = 10^{-4} and 10^{-11} for single and double recision, respectively.

5. THE COMPUTER PROGRAMS

A computer program entitled PC1L1POL and its subsidiary routine LONEPOL are written in FORTRAN 77 for the present algorithm. The algorithm is tested on a number of problems in single precision calculation on the VAX 11/780 computer.

6. NUMERICAL RESULTS

In order to compare the results of the present algorithm with those of related ones, such as of Abdelmalek (3) for the linear L_1 piecewise approximation and of Kurozumi and Davis (9) for the Chebyshev polygonal approximation, we chose the same curves used in these two references.

The curve of the first example is originally that of Figure 5 in Tomek (14). It is digitized with equal x-intervals into 28 points. The other three examples are those given in Kurozumi and Davis (9) and the digitized points are not equally spaced. The results of the present algorithm are given in Figs. 1-4. Compare these results with those of Abdelmalek (3) and Kurozumi and Davis (9).

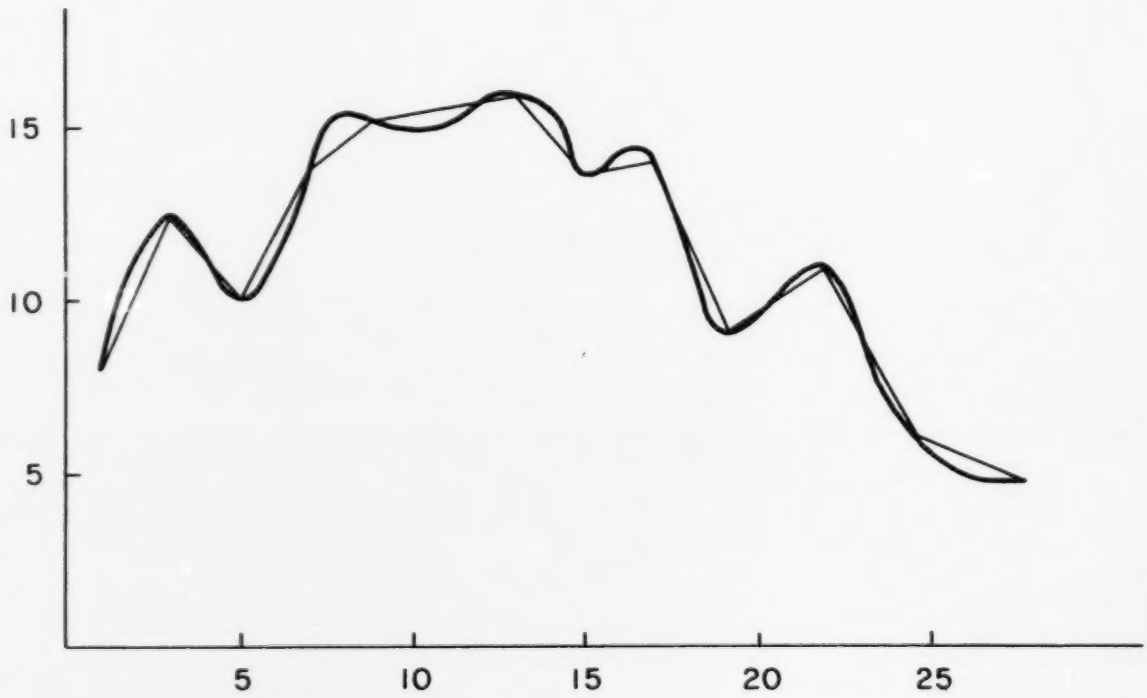

Figure 1. Polygonal approximation for example 1, the L_1 norm in any segment is <1.0.

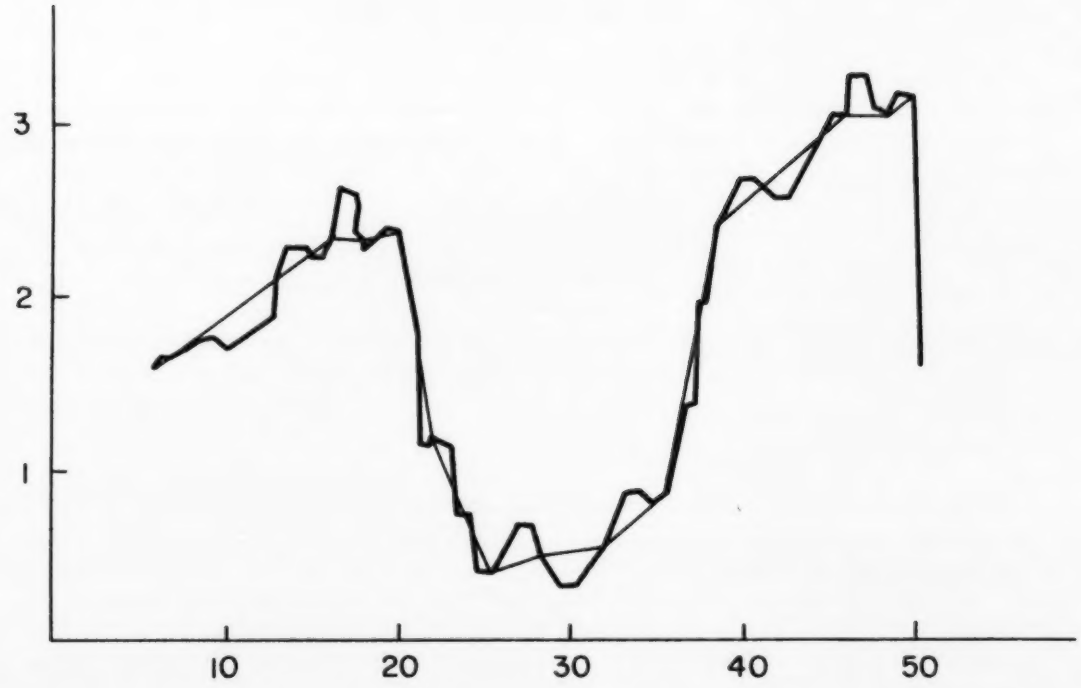

Figure 2. Polygonal approximation for a wave form, the L_1 norm in any segment is <1.0 .

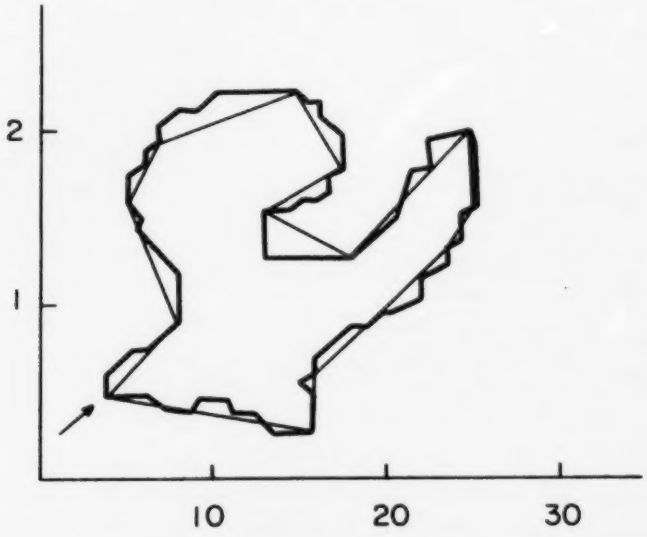

Figure 3. Polygonal approximation for a contour, the L_1 norm in any any segment is <0.5 .

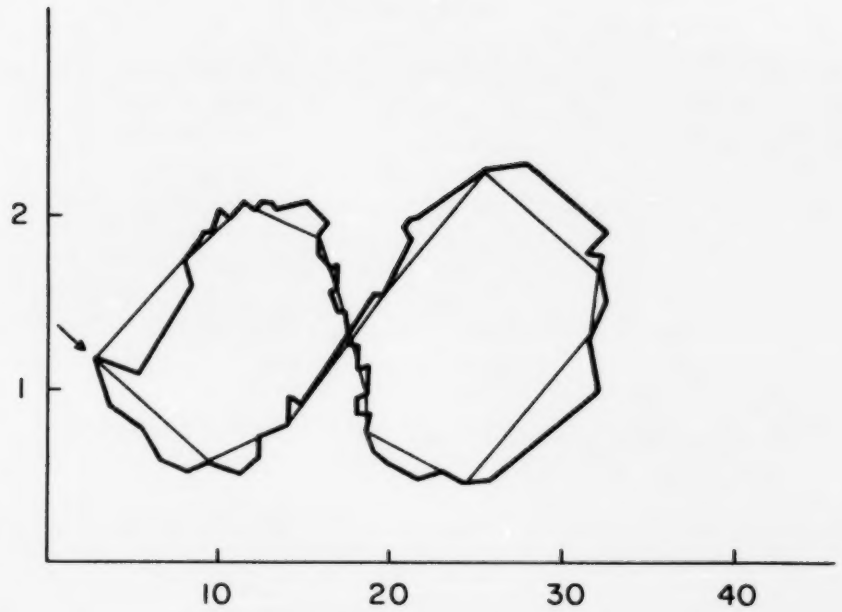

Figure 4. Polygonal approximation for a self intersecting curve, the L_1 norm in any segment is <0.8.

7. COMMENTS AND CONCLUSION

Simple arithmetic operations count shows that if segment j consists of N points, the algorithm requires ON^2 multiplications/divisions to calculate the line joining its two end points.

In calculating the polygonal approximation of planar curves in the Chebyshev norm, Kurozumi and Davis (9), for example, made sure that the residuals are calculated perpendicular to the approximating straight lines. This requirement is not needed in the present algorithm, particularly when the given curve is digitized with equal x-intervals. The reason is that the L_1 error norm for a digitized curve segment with equal x-intervals is approximately proportional to the area between the segment and the approximating straight line. This area is approximately the same when the residuals are measured in the y-direction with the x projection of the approximating line as the base, as is the case with our algorithm, or when measured perpendicular to the approximating straight line and the straight line itself as the base.

Finally, in contrast to our algorithm, in the algorithm of Johnson and Vogt (5), the given curve is not digitized, it has to be convex and the number of vertices are given.

ACKNOWLEDGMENTS

I wish to thank Catherine Merritt for reading the manuscript and for interesting discussions. I also thank Denis Saumure for his help in drawing the illustrations. Finally, I wish to thank the staff of the Editorial Office of the Division of Electrical Engineering of the NRC for their assistance in the publication of this and my other reports.

REFERENCES

1. T. Pavlidis. The use of algorithms of piecewise approximations for picture processing applications. ACM Trans. Math. Software. 2: 305-321; 1976.

2. J.R. Rice and J.S. White. Norms for smoothing and estimation. SIAM Rev. 6: 243-256; 1964.

3. N.N. Abdelmalek. Piecewise linear L_1 approximation of planar curves. Int. J. Syst. Sci. 16: 447-455; 1985. NRC 24425.

4. N.N. Abdelmalek. Programs for piecewise linear L_1 approximation of planar curves. NRC/ERB-965; May 1984. NRC 23411.

5. H.H. Johnson and A. Vogt. A geometric method for approximating convex arc. SIAM J. Appl. Math. 38: 317-325; 1980.

6. N.N. Abdelmalek. Polygonal approximation of planar curves in the L_1 norm. Int. J. Syst. Sci. In press.

7. G. Hadley. Linear Programming. Addison-Wesley, Reading, MA. 1962.

8. N.N. Abdelmalek. Piecewise linear Chebyshev approximation of planar curves. Int. J. Syst. Sci. 14: 425-435; 1983. NRC 22642.

9. Y. Kurozumi and W.A. Davis. Polygonal approximation by minimax method. Comput. Graph. Image Process. 19: 248-264; 1982.

10. J. Sklansky and V. Gonzalez. Fast polygonal approximation of digitized curves. Pattern Recog. 12: 327-331; 1980.

11. F. Gritzali and G. Papakonstantinou. A fast piecewise linear approximation algorithm. Signal Process. 5: 221-227; 1983.

12. J.R. Rice. The Approximation of Functions, Part I. Addison-Wesley, Reading, MA. 1964.

13. N.N. Abdelmalek. On the discrete linear L_1 approximation and L_1 solutions of overdetermined linear equations. J. Approx. Theory, 11: 38-53; 1974.

14. I. Tomek. Two algorithms for piecewise-linear continuous approximation of functions of one variable. IEEE Trans. Comput. C-23: 445-448; 1974.

APPENDIX

```
C                    "DRIVER DR1L1POL"
C
C     THIS PROGRAM IS A DRIVER FOR THE  SUBROUTINE  PC1L1POL
C WHICH CALCULATES THE POLYGONAL STRAIGHT LINE APPROXIMATION
C IN  THE  L-ONE  NORM  OF  A  GIVEN  DATA  POINT  SET {X,Y}
C RESULTING FROM THE DISCRETIZATION OF A PLANE CURVE  Y=F(X)
C OF ARBITRARY SHAPE.
C
C     THE APPROXIMATION BY SUBROUTINE PC1L1POL IS  SUCH THAT
C THE  APPROXIMATING  STRAIGHT  LINE  INTERPOLATES  THE  END
C POINTS OF THE SEGMENT AT HAND OF THE GIVEN CURVE  AND ALSO
C SUCH THAT THE L-ONE NORM FOR  ANY SEGMENT IS NOT TO EXCEED
C A  PRE-ASSIGNED  TELERANCE  'ENORM'.   THE   APPROXIMATING
C STRAIGHT  LINES ARE CONNECTED IN THE FORM OF A POLYGON.
C
C     THE  MAXIMUM  NUMBER  OF  POINTS  {X,Y}  IS  250.  THE
C THE MAXIMUM NUMBER OF PIECES  (SEGMENTS) IN THE  PIECEWISE
C APPROXIMATION IS 50.  THESE  NUMBERS ARE THE VALUES OF THE
C PARAMETERS 'NN' AND  'KK' RESPECTIVELY IN THE SUBROUTINES.
C THESE NUMBERS MAY  BE  EASILY CHANGED  IN  THE   DIMENSION
C STATEMENTS AS WELL AS IN THE STATEMENTS 'NN=250 AND KK=50'
C IN THIS DRIVER.
C
C     THE CALCULATION IS IN SINGLE PRECISION. IT MAY BE CON-
C VERTED TO DOUBLE PRECISION FOR THE IBM AND THE VAX COMPUT-
C ERS BY SIMPLY REPLACING THE STATEMENT 'DOUBLE PRECISION S'
C BY THE STATEMENT   'IMPLICIT REAL*8(A-H,O-Z)'   IN EACH OF
C THE DRIVER AND THE TWO SUBROUTINES.   SEE ALSO THE COMMENT
C ABOVE THE  STATEMENT 'EPS=1.E-04' IN THIS DRIVER.
C
C     THIS DRIVER CONTAINS 4 EXAMPLES.
C
      REAL*4 TIME,ORIGIN
      DOUBLE PRECISION S
      DIMENSION CT(2,250),F(250),R(250),A(2),BINV(2,2),
     1ICBAS(2)
      DIMENSION CTN(2,250),FN(250)
      DIMENSION RP(250),AP(50,2),ZP(50),IXL(50),V(2)
      DIMENSION FC(28),FD(57),FE(62),FG(64),
     *XD(57),XE(62),XG(64)
C
      DATA FC/8.0,11.0,12.4,11.2,10.1,11.0,13.8,15.4,15.1,
     115.0,15.1,15.8,16.0,15.6,13.5,14.3,14.1,10.6,8.9,9.6,
     210.5,10.8,9.0,6.8,5.8,5.0,4.8,4.9/
C
      DATA XD/6.,6.2,7.,8.5,9.5,10.2,12.9,13.1,13.6,14.8,
     115.,15.9,16.4,16.8,17.8,17.8,18.,18.1,19.3,20.1,
     221.,21.5,21.9,22.1,23.2,23.5,24.2,24.5,25.5,26.8,
     328.,28.2,29.2,30.2,32.,33.1,34.,34.9,35.7,36.6,
     437.3,37.4,37.8,38.6,39.7,40.8,41.9,42.8,45.1,46.,
     546.,47.,47.6,48.1,49.,49.9,50.2/
```

```
C
      DATA FD/1.6,1.65,1.65,1.75,1.78,1.7,1.88,2.12,2.28,
     12.28,2.22,2.23,2.38,2.63,2.6,2.38,2.32,2.28,2.4,2.38,
     21.8,1.15,1.12,1.19,1.13,.72,.72,.4,.39,.68,
     3.67,.5,.31,.31,.51,.86,.87,.79,.85,1.35,
     41.35,1.95,1.95,2.4,2.67,2.68,2.58,2.57,3.03,3.05,
     53.33,3.36,3.08,3.03,3.15,3.12,1.58/
C
      DATA XE/3.8,6.6,7.,8.8,9.1,10.9,11.1,12.3,13.5,15.8,
     115.8,14.9,15.8,15.9,18.,19.,19.7,20.4,22.,22.,
     223.5,23.5,24.1,24.1,25.2,25.1,24.8,22.2,22.5,21.3,
     320.5,18.,13.,13.,14.5,15.,15.9,16.6,16.6,17.8,
     417.8,16.,16.,15.3,14.9,10.,9.,8.,6.9,7.,
     56.,6.,5.,5.,5.8,5.7,8.,8.,6.,5.1,
     63.8,3.8/
C
      DATA FE/.48,.48,.4,.39,.47,.46,.38,.38,.27,.28,
     1.5,.55,.6,.7,.89,.89,.96,.96,1.02,1.16,
     21.23,1.31,1.38,1.51,1.57,1.92,2.,1.95,1.79,1.77,
     31.5,1.26,1.26,1.53,1.55,1.6,1.6,1.65,1.72,1.78,
     41.98,2.1,2.17,2.17,2.23,2.22,2.11,2.11,2.02,1.95,
     51.89,1.8,1.74,1.59,1.5,1.41,1.19,.9,.73,.72,
     6.6,.48/
C
      DATA XG/2.8,3.9,5.6,6.7,8.2,9.5,11.3,12.4,12.4,14.1,
     114.1,15.,19.,19.8,21.,20.9,25.2,28.,32.5,31.6,
     232.5,32.1,32.6,31.5,32.,30.1,26.1,24.2,23.1,21.7,
     319.9,19.,18.7,19.,18.1,18.1,18.7,18.9,18.1,18.1,
     417.6,17.4,16.9,16.8,17.,17.1,16.5,15.9,16.,16.3,
     515.2,13.3,13.,12.5,12.1,11.5,10.5,10.,9.5,9.,
     68.1,8.5,5.5,2.8/
C
      DATA FG/1.18,.9,.8,.6,.51,.6,.52,.6,.73,.8,
     1.98,.92,1.58,1.58,1.89,1.95,2.3,2.32,1.92,1.8,
     21.8,1.7,1.52,1.3,1.02,.82,.5,.48,.52,.5,
     3.6,.66,.8,.88,.88,.99,.99,1.15,1.13,1.38,
     41.38,1.46,1.46,1.58,1.58,1.73,1.7,1.8,1.9,1.98,
     52.1,2.08,2.09,2.09,2.03,2.1,2.,2.05,1.92,1.9,
     61.75,1.6,1.1,1.18/
C
  101 FORMAT(10H1 EXAMPLE ,I3,5H  N= ,I3,5H  M= ,I3,
     *9H  ENORM= ,F10.5)
  102 FORMAT(40H0F(J)                                          )
  103 FORMAT(40H0TRANSPOSE OF MATRIX C                         )
  104 FORMAT(17H0EXECUTION TIME =,F10.5,13HSECONDS             )
  105 FORMAT(30H0  IXL(J):J=1,NPIECE                           )
  107 FORMAT(25H0    I,  AP                                    )
  108 FORMAT(25H0    ZP                                        )
  109 FORMAT(25H0    RP                                        )
  110 FORMAT(1H ,10F7.3)
  111 FORMAT(1H ,10I5)
```

```
  112 FORMAT(1H ,I5,2X,8F8.3)
      ORIGIN=0.0
      NN = 250
      KK = 50
      MM = 2
      M = 2
C FOR DOUBLE PRECISION  EPS=1.E-11 INSTEAD OF  EPS=1.E-04.
      EPS=1.E-04
      IEXMPL = 0
    1 IEXMPL=IEXMPL+1
      GO TO (2,4,6,8, 100), IEXMPL
C
C EXAMPLE 1. IT IS THAT OF FIGURE 5 IN I. TOMEK, TWO ALGOR-
C ITHMS FOR PIECEWISE-LINEAR  CONTINUOUS  APPROXIMATION  OF
C FUNCTIONS OF ONE  VARIABLE.  IEEE TRANS. COMPUTERS, C-23,
C 445-448; 1974.
    2 N=28
      DO 3 J=1,N
        CT(1,J) = 1.0
        CT(2,J) = FLOAT(J)
    3   F(J) = FD(J)
      ENORM = 1.0
      GO TO 80
C
C EXAMPLE 2. EXAMPLES 2, 3 AND 4  ARE THOSE OF  Y. KUROZUMI
C AND  W.A. DAVISavis,  POLYGONAL  APPROXIMATION BY MINIMAX
C METHOD.  COMPUTER  GRAPHICS AND IMAGE PROCESSING 19: 248-
C 264; 1982.
    4 N=57
      DO 5 J=1,N
        CT(1,J)=1.0
        CT(2,J)=XD(J)
    5   F(J)=FD(J)
      ENORM=0.65
      GO TO 80
C
C EXAMPLE 3.
    6 N = 62
      DO 7 J=1,N
        CT(1,J) = 1.0
        CT(2,J) = XE(J)
    7   F(J) = FE(J)
      ENORM=0.5
      GO TO 80
C
C EXAMPLE 4.
    8 N=64
      DO 9 J=1,N
        CT(1,J) = 1.0
        CT(2,J) = XG(J)
    9   F(J) = FG(J)
```

```
      ENORM=0.8
      GO TO 80
   80 CONTINUE
C
      WRITE(3,101)IEXMPL,N,M,ENORM
      WRITE(3,102)
      WRITE(3,110)(F(J),J=1,N)
      WRITE(3,103)
      DO 91 I=1,M
   91   WRITE(3,110)(CT(I,J),J=1,N)
C SUBROUTINES SETCLK AND RDCLK ARE FOR MEASURING THE CPU
C TIME 'TIME' USED BY THE SUBROUTINE AT HAND; 'PC1L1POL'
C IN OUR CASE.
      CALL SETCLK(ORIGIN)
C
      CALL PC1L1POL(M,N,MM,NN,KK,CT,F,EPS,ICBAS,BINV,
     1ITER,R,A,Z,IND,ENORM,CTN,RP,AP,ZP,IXL,V,NPIECE,
     2KASE)
C
      CALL RDCLK(TIME,ORIGIN)
      WRITE(3,104)TIME
      WRITE(3,105)
      WRITE(3,111)(IXL(J),J=1,NPIECE)
      WRITE(3,107)
      DO 92 I=1,NPIECE
   92   WRITE(3,110)(AP(I,J),J=1,M)
      WRITE(3,108)
      WRITE(3,110)(ZP(I),I=1,NPIECE)
      WRITE(3,109)
      WRITE(3,110)(RP(J),J=1,N)
      GO TO 1
  100 STOP
      END
```

```
      SUBROUTINE PC1L1POL(M,N,MM,NN,KK,CT,F,EPS,ICBAS,
     1BINV,ITER,R,A,Z,IND,ENORM,CTN,RP,AP,ZP,IXL,V,NPIECE,
     2KASE)
C
C     GIVEN IS A SET OF POINTS {X,Y} RESULTING FROM DIGITIZ-
C ING  A  GIVEN  PLANE  CURVE OF ARBITRARY SHAPS OF THE FORM
C Y=F(X). ONE FORMS  THE  OVERDETERMINED  SYSTEM  OF  LINEAR
C EQUATIONS
C                              C*A=F.
C C IS AN N BY 2 MATRIX. EACH ELEMENT OF THE FIRST COLUMN OF
C C IS 1. THE SECOND COLUMN OF C CONTAINS THE  X-COORDINATES
C OF  THE POINTS  {X,Y}  IN  A  SEQUENTIAL ORDER. F IS  AN N
C VECTOR AND CONTAINS THE Y-COORDINATES OF THE SET  {X,Y} IN
C THE  SAME  ORDER AS THE X-COORDINATES IN THE SECOND COLUMN
C OF MATRIX C.
C
C     THIS PROGRAM IS A SUBROUTINE FOR CALCULATING  A LINEAR
C POLYGONAL STRAIGHT LINE PIECEWISE L-ONE  APPROXIMATION  TO
C THE DISCRETE  POINT SET {X,Y}.  THE APPROXIMATION IS  SUCH
C THAT THE L-ONE ERROR NORM FOR ANY SEGMENT IS NOT TO EXCEED
C A PRE-ASSIGNED TOLERANCE 'ENORM'.   THE NUMBER OF SEGMENTS
C IS NOT KNOWN BEFORE HAND.
C
C     THIS  SUBROUTINE  USES  LINEAR PROGRAMMING  TECHNIQUES
C AND MAKES USE OF THE INTERPOLATING PROPETY OF THE DISCRETE
C LINEAR L-ONE APPROXIMATION.
C
C     THE SUBROUTINE IS WRITTEN IN SINGLE  PRECISION. IT MAY
C BE  CONVERTED  TO  DOUBLE PRECISION FOR THE IBM AND VAX BY
C REPLACING THE STATE MENT   'DOUBLE PRECISION S'  BY    THE
C STATEMENT 'IMPLICIT REAL*8(A-H,O-Z)'.
C
C     FINALLY THIS SUBROUTINE CALLS A  SUBSIDIARY SUBROUTINE
C NAMED 'LONEPOL', FOR EXECUTING THE RESULTS.
C
C     THE INPUT DATA TO THE SUBROUTINE.
C M       THE NUMBER OF TERMS IN THE  APPROXIMATING FUNCTION.
C         FOR THE STRAIGHT LINE Y=A(1)+A(2)*X, M=2.
C N       THE  TOTAL  NUMBER OF  POINTS {X,Y} TO BE PIECEWISE
C         APPROXIMATED.
C MM      AN INTEGER GREATER THAN OR EQUAL TO M.
C NN      AN INTEGER GREATER THAN OR EQUAL TO N.
C KK      AN INTEGER  LARGE ENOUGH SUCH THAT THE TOLAL NUMBER
C         OF SEGMENTS IS TO BE LE.KK .
C CT      A MATRIX OF DIMENSIONS MM BY NN.  ITS  FIRST M ROWS
C         AND  FIRST  N  COLUMNS  CONTAIN  THE  TRANSPOSE  OF
C         MATRIX C OF THE SYSTEM C*A=F.  MATRIX  CT  DOES NOT
C         CHANGE IN THE COMPUTATION.
C F       AN NN VECTOR.  ITS  FIRST  N  ELEMENTS  CONTAIN THE
C         R.H.S. OF THE SYSTEM C*A=F.  I.E, VECTOR F CONTAINS
C         THE Y-COORDINATES OF THE GIVEN  POINT  SET.  VECTOR
```

```
C         F DOES NOT CHANGE IN THE COMPUTATION.
C EPS     A SPECIFIED TOLERANCE SUCH THAT A CALCULATED
C         NUMBER Z IS CONSIDERED ZERO IF |Z|<EPS. FOR THE
C         VAX 11/780 COMPUTER, EPS IS USUALLY TAKEN 1.E-04
C         AND 1.E-11 FOR SINGLE AND DOUBLE PRECISION
C         RESPECTIVELY.
C ENORM   A REAL NUMBER; A PRE-ASSIGNED TOLERANCE, SUCH THAT
C         THE L-ONE NORM FOR ANY SEGMENT IS .LE.ENORM .
C
C     THE OUTPUT OF THE SUBROUTINE.
C NPIECE AN INTEGER WHICH IS THE NUMBER OF SEGMENTS OR
C         PIECES OF THE APPROXIMATION; NPIECE.LE.KK .
C IXL     A KK-VECTOR. ITS FIRST 'NPIECE' ELEMENTS CONTAIN
C         THE INDICES OF THE FIRST ELEMENTS OF THE NPIECE
C         SEGMENTS. FOR EXAMPLE IF IXL=(1,5,12,22, ... ),
C         THEN THE FIRST SEGMENT CONTAINS POINTS 1 TO 5, THE
C         SECOND SEGMENT CONTAINS POINTS 5 TO 12, THE THIRD
C         SEGMENT CONTAINS POINTS 12 TO 22 ...,ETC.
C RP      AN NN VECTOR. ITS FIRST N ELEMENTS CONTAIN THE
C         RESIDUAL VALUES OF THE N GIVEN POINTS OF THE GIVEN
C         SET OF POINTS {X,Y}. R=C*A-F.
C AP      AN KK BY MM MATRIX. THE FIRST M ELEMENTS OF ITS
C         FIRST ROW CONTAIN THE ELEMENTS OF THE SOLUTION
C         VECTOR A* FOR THE FIRST SEGMENT. THE FIRST M
C         ELEMENTS OF ITS SECOND ROW CONTAIN THE SOLUTION
C         VECTOR A* FOR THE SECOND SEGMENT AND SO ON. IF THE
C         FIRST M ELEMENTS OF ROW J SAY ARE ALL ZEROS, THIS
C         INDICATES THAT VECTOR A* OF ROW J IS NOT
C         CALCULATED BECAUSE THE NUMBER OF POINTS OF SEGMENT
C         J IS < M AND THERE IS A PERFECT FIT FOR SEGMENT J.
C ZP      AN KK VECTOR. ITS FIRST NPIECE ELEMENTS CONTAIN
C         THE NPIECE OPTIMUM VALUES OF THE L-ONE NORMS FOR
C         THE NPIECE SEGMENTS. IF ZP(J)=0.0, THIS INDICATES
C         THAT THERE IS A PERFECT FIT FOR SEGMENT J.
C
C     THE MEANING OF THE OTHER PARAMETERS.
C CTN     AN MM BY NN MATRIX. ON EXIT, THE FIRST M ROWS AND
C         THE FIRST N COLUMNS OF CTN CONTAIN THE FINAL
C         SIMPLEX TABLEAUX FOR ALL THE SEGMENTS.
C ICBAS,BINV,ITER,R,A,Z,IND:                          ARE
C         PARAMETRS TO BE PASSED TO THE AUXILIARY SUBROUTINE
C         'LONEPOL' AND THEY HAVE THE SAME MEANINGS AS IN THE
C         SUBROUTINE LONEPOL.
C V       AN MM WORKING VECTOR.
C
      DOUBLE PRECISION S
      DIMENSION CT(MM,NN),F(NN),R(NN),A(MM),BINV(MM,MM),
     1ICBAS(MM),V(MM)
      DIMENSION CTN(MM,NN),RP(NN),AP(KK,MM),ZP(KK),IXL(KK)
C
      NPIECE=1
```

```
      IS=1
C
C IS MEANS I(START)
C IE MEANS I(END)
C     KASE.EQ.0 INDICATES THE STAR OF THE ALGORITHM.
    1 KASE=0
      ZP(NPIECE)=0.0
      IE=IS+M-1
    2 DO 3 J=IS,IE
    3   RP(J)=0.0
      DO 4 I=1,M
    4   AP(NPIECE,I)=0.0
      IF(IE.GT.N) IE=N
      IXL(NPIECE)=IS
      NU=IE-IS+1
      IF(NU.LT.M) GO TO 17
C
      IF(KASE.NE.0) GO TO 6
      DO 5 J=IS,IE
        DO 5 I=1,M
    5     CTN(I,J)=CT(I,J)
C
    6 CALL LONEPOL(M,N,IS,IE,MM,NN,CTN,F,EPS,ICBAS,
     *BINV,ITER,R,A,Z,IND,KASE)
C
      IF(Z.GT.ENORM) GO TO 16
      IF(IND.EQ.0) GO TO 7
      IE=IE+1
      GO TO 2
C
    7 ZP(NPIECE)=Z
      DO 8 J=1,M
    8   AP(NPIECE,J)=A(J)
      DO 9 J=IS,IE
    9   RP(J)=R(J)
      KASE=-1
   10 JE=IE+1
      IF(JE.GT.N) GO TO 17
      DO 11 I=1,M
   11   V(I)=CT(I,JE)
C
C CALCULATING THE MARGINAL COSTS (THE RESIDUALS)
      DO 13 I=1,M
        S=0.0
        DO 12 K=1,M
   12     S=S+BINV(I,K)*V(K)
   13   CTN(I,JE)=S
      SA=-F(JE)
      DO 14 I=1,M
   14     SA=SA+A(I)*CT(I,JE)
      R(JE)=SA
```

```
20
      PIV=CTN(2,JE)
      IF(PIV.LT.0.0) PIV=-PIV
      IF(PIV.GT.EPS) GO TO 15
      IE=IE+1
      GO TO 10
C
   15 IE=JE
      GO TO 6
   16 ISO=IS
      IS=IE-1
      IF(IS.EQ.ISO) WRITE(3,125)
      IF(IS.EQ.ISO) GO TO 17
      NPIECE=NPIECE+1
      GO TO 1
  125 FORMAT(40HO ENDLESS LOOP,  INCREASE VALUE OF ENORM)
C
   17 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE LONEPOL(M,N,N1,N2,MM,NN,CT,F,EPS,ICBAS,
     *BINV,ITER,R,A,Z,IND,KASE)
C
C     LET US BE GIVEN  AN  OVERDETERMINED  SYSTEM  OF LINEAR
C EQUATIONS OF THE FORM
C                              C*A=F
C WHERE C IS A GIVEN  REAL  N BY 2  MATRIX OF RANK LE.M.LE.N
C AND F IS A GIVEN REAL N-VECTOR. A=(A(1),A(2)).
C     THIS  SUBROUTINE  SOLVES  THE FIRST AND THE LAST EQUA-
C TIONS  OF  THE  OVERDETERMINED  SYSTEM  C*A=F  FOR THE TWO
C UNKNOWNS (A(1),A(2)). IT USES A LINEAR PROGRAMMING  METHOD
C SUCH  THAT  IF AN EXTRA  EQUATION IS   ADDED TO THE SYSTEM
C C*A=F,  THE SUBROUTINE SOLVES THE FIRST  AND THE NEW  LAST
C EQUATION IN ONE SIMPLEX STEP. IT ALSO CALCULATES THE L-ONE
C NORM Z OF THE RESIDUAL VECTOR R FROM THE SIMPLEX TABLEAU.
C              Z=|R(1)|+|R(2)|+ ... +|R(N)|,
C WHERE R(I) IS THE ITH RESIDUAL AND IS GIVEN BY
C   R(I)=C(I,1)*A(1)+C(I,2)*A(2)+ ... +C(I,M)*A(M)-F(I).
C
C     THE INPUT DATA TO THIS SUBROUTINE.
C M       THE NUMBER OF COLUMNS OF MATRIX C (M=2).
C N       THE NUMBER OF ROWS OF MATRIX C.
C MM      AN INTEGER GREATER THAN OR EQUAL TO M.
C NN      AN INTEGER GREATER THAN OR EQUAL TO N.
C CT      A MATRIX OF DIMENSIONS  MM  BY NN. ITS FIRST M ROWS
C         AND FIRST N COLUMNS CONTAIN THE TRANSPOSE OF MATRIX
C         C OF THE SYSTEM C*A=F.
C F       AN NN-VECTOR. ITS FIRST N ELEMENTS CONTAIN VECTOR F
C         OF THE SYSTEM C*A=F.
C EPS     A  SPECIFIED  TOLERANCE  SUCH  THAT  A   CALCULATED
C         NUMBER  Z  IS  CONSIDERED  ZERO IF |Z|<EPS. FOR THE
C         VAX 11/780 COMPUTER, EPS  IS USUALLY  TAKEN  1.E-04
C         AND   1.E-11   FOR  SINGLE  AND  DOUBLE   PRECISION
C         RESPECTIVELY.
C
C     THE RESULTS OF THE PROBLEM.
C ITER    THE NUMBER OF ITERATIONS OR THE NUMBER OF TIMES THE
C         SIMPLEX TABLEAU IS CHANGED BY A GAUSS-JORDAN STEP.
C A       AN MM-VECTOR. ON EXIT, ITS FIRST M ELEMENTS CONTAIN
C         THE SOLUTION OF THE PROBLEM FOR THE SEGMENT AT HAND
C R       AN NN-VECTOR.  ITS  FIRST  N  ELEMENTS  CONTAIN THE
C         RESIDUAL VECTOR R=(C*A-F) FOR THE SEGMENT AT HAND.
C Z       THE L-ONE NORM OF THE RESIDUAL VECTOR R.
C IND     A  RETURN INDICATOR.  IND=0  INDICATES  A  SOLUTION
C         VECTOR  A*  IS OBTAINED.  IND=-1 INDICATES THAT THE
C         CALCULATION  IS  TERMINATED  PREMATURELY DUE  TO THE
C         ILL-CONDITIONING  OF  MATRIX  C  FOR THE SEGMENT AT
C         HAND.
C
C     THE MEANING OF THE OTHER PARAMETERS.
C N1      INDEX OF THE FIRST POINT OF THE SEGMENT AT HAND.
```

```
C N2      INDEX OF THE LAST POINT OF THE SEGMENT AT HAND.
C ICBAS   AN MM-VECTOR.  ITS FIRST M ELEMENTS ARE THE INDICES
C         OF THE COLUMNS OF CT WHICH FORM THE BASIS.
C BINV    AN MM SQUARE MATRIX.  ITS  FIRST M ROWS AND FIRST M
C         COLUMNS CONTAIN THE INVERSE OF THE BASIS  MATRIX IN
C         THE LINEAR PROGRAMMING PROBLEM.
C KASE    AN  INTEGER SUCH THAT FOR  KASE=0  THE   SUBROUTINE
C         SOLVES THE FIRST AND LAST EQUATIONS OF SYSTEM C*A=F
C         FOR THE FIRST TIME.  FOR KASE.NE.0, THE  SUBROUTINE
C         SOLVES THE FIRST AND THE (NEW) LAST EQUATION OF THE
C         SYSTEM C*A=F IN ONE SIMPLEX STEP.
C
      DOUBLE PRECISION S,SA,SB
      DIMENSION CT(MM,NN),F(NN),R(NN),A(MM),BINV(MM,MM),
     *ICBAS(MM)
C
      IND=0
      ITER=0
      IF(KASE.NE.0) GO TO 4
C
C KASE.NE.0  INDICATES THAT THE SEGMENT AT HAND IS  ENLARGED
C BY ADDING TO ITS END A EXTRA POINT.
C
    1 DO 3 J=1,M
        A(J)=0.
        ICBAS(J)=0
        DO 2 I=1,M
    2     BINV(I,J)=0.
    3   BINV(J,J)=1.
      z=0.
      JIN=N1
      IOUT=1
      GO TO 9
    4 IOUT=2
      JIN=N2
      PIVOT=CT(IOUT,JIN)
      PIV=PIVOT
      IF(PIV.LT.0.0) PIV=-PIV
C DETECTION OF RANK DEFICIENCY.
      IF(PIV.GT.EPS) GO TO 9
C MATRIX B IS SINGULAR
      IND=-1
      GO TO 20
C CALCULATING THE RESIDUALS (MARGINAL COSTS).
    5 DO 8 J=N1,N2
        R(J)=0.0
        DO 6 I=1,M
          IF(J.EQ.ICBAS(I)) GO TO 8
    6     CONTINUE
          S=-F(J)
        DO 7 I=1,M
```

```
            K=ICBAS(I)
            SA=F(K)
            SB=CT(I,J)
    7       S=S+SA*SB
          R(J)=S
    8   CONTINUE
        GO TO 16
C A GAUSS-JORDAN ELIMINATION STEP.
    9   PIVOT=CT(IOUT,JIN)
        ICBAS(IOUT)=JIN
        DO 10 J=N1,N2
   10     CT(IOUT,J)=CT(IOUT,J)/PIVOT
        DO 11 J=1,M
   11     BINV(IOUT,J)=BINV(IOUT,J)/PIVOT
        DO 14 I=1,M
          IF(I.EQ.IOUT) GO TO 14
          E=CT(I,JIN)
          DO 12 J=N1,N2
   12       CT(I,J)=CT(I,J)-E*CT(IOUT,J)
          DO 13 J=1,M
   13       BINV(I,J)=BINV(I,J)-E*BINV(IOUT,J)
   14     CONTINUE
        IF(IOUT.EQ.1) GO TO 4
        ITER=ITER+1
        IF(KASE.EQ.0) GO TO 5
        E=R(JIN)
        DO 15 J=N1,N2
   15     R(J)=R(J)-E*CT(IOUT,J)
C CALCULATING THE ANSWER OF THE L-ONE PROBLEM.
   16   DO 18 J=1,M
          S=0.
          DO 17 I=1,M
            K=ICBAS(I)
            SA=F(K)
            SB=BINV(I,J)
   17       S=S+SA*SB
   18     A(J)=S
        S=0.
        DO 19 J=N1,N2
          SA=R(J)
          IF(SA.LT.0.0) SA=-SA
   19     S=S+SA
        Z=S
   20   CONTINUE
        RETURN
        END
```